

Algoritmos y Estructuras de Datos II

Primer parcial – Sábado 5 de mayo de 2018

- El parcial es a libro abierto.
- Cada ejercicio debe entregarse en hojas separadas.
- Incluir en cada hoja el número de orden asignado, número de hoja, apellido y nombre.
- Antes de entregar, remover los "pelitos" del borde de las hojas, si hubiere.
- Cada ejercicio se calificará con Perfecto, Aprobado, Regular, o Insuficiente. El parcial estará aprobado si: (el ejercicio 1 tiene al menos A) y (el ejercicio 3 tiene al menos una A o (el ejercicio 3 tiene una R y el 2 al menos una A)).

Ej. 1. Especificación

El Centro Comunitario de Carpinteros (CCC) brinda a la gente la posibilidad de aprender carpintería a nivel profesional. El curso de capacitación tiene 10 niveles y se ofrece al público en forma gratuita. Para ello, el CCC emplea como docentes (voluntarios) a los mismos carpinteros que se formaron o se están formando en el CCC. Se desea modelar el CCC utilizando un TAD, siguiendo las características descriptas a continuación.

Inicialmente, el CCC cuenta con un conjunto de maestros carpinteros capaces de enseñar cualquiera de los niveles del curso. En cualquier momento, una persona ajena al CCC puede iniciar el curso empezando por el nivel 1. En ese momento se le asigna un *tutor*, el cual podrá ser uno de los maestros del CCC o bien alguna de las otras personas que estén cursando niveles posteriores al nivel 1 (aun no está definido el procedimiento a seguir para realizar esta elección). La persona se convierte entonces en *aprendiz* de su tutor. La responsabilidad del tutor es realizar un seguimiento de sus aprendices y acompañarlos en su aprendizaje. Cuando el tutor detecta que un aprendiz ya comprendió lo necesario del nivel en el que está, se avanza a este último al siguiente nivel. No es deseable que una persona tenga aprendices de su mismo nivel (ni de niveles superiores obviamente), con lo cual, si al avanzar de nivel, un aprendiz alcanza el nivel de su tutor, automáticamente se le asigna un nuevo tutor siguiendo las mismas reglas que cuando ingresó al CCC. Vale aclarar que si el aprendiz completó el nivel 10, no se le asigna ningún tutor (ya que no sería posible), y en este caso el aprendiz pasa a ser parte del conjunto de maestros carpinteros del CCC.

En cualquier momento, una persona p (ya sea aprendiz o maestro carpintero) puede desertar del CCC. Si p tenía aprendices en ese momento, éstos pasan a ser aprendices del tutor de p . En el caso en que p fuese un maestro carpintero, sus aprendices pasarán a ser aprendices de algún otro maestro carpintero (aunque no está definido aun cómo se lo elegirá). Es importante notar que el CCC nunca puede quedarse sin maestros carpinteros. Una persona que deserta del CCC, no puede volver a ser miembro del mismo.

Modelar con un TAD el CCC descripto teniendo en cuenta además que, dado un miembro, interesa saber quiénes son actualmente sus *descendientes académicos*. Los *descendientes académicos* de un miembro son todos sus aprendices, y los aprendices de sus aprendices, y los aprendices de estos últimos, y así sucesivamente.

Ej. 2. Complejidad

Discutir la veracidad de las siguientes afirmaciones, justificando adecuadamente en cada caso:

1. Si $f(n) \in O(n)$ entonces $2^{f(n)} \in O(2^n)$.
2. Siendo $f \cdot g$ la función $(f \cdot g)(x) = f(x)g(x)$, vale que $f \cdot g \in \Theta(\max\{f, g\}^2)$.
3. La complejidad temporal del *peor caso* del Algoritmo 1 es $\Theta(n^2)$.
4. La complejidad temporal del *mejor caso* del Algoritmo 1 es $O(n)$.

Algoritmo 1 Suma de algunos productos menores que n^2

```

1: function SUMAALGUNOSPRODUCTOS(in n: entero)
2:   acum = 0
3:   for i := 1 ... n do  $\Theta(n)$ 
4:     if i ≤ 100 then
5:       for j := 1 ... n do  $\Theta(n)$ 
6:         acum := acum + i*j
7:       end for
8:     end if
9:   end for
10:  return acum
11: end function

```

Ej. 3. Diseño

Técnicos a Domicilio (TaD), provee servicio técnico para hogares y empresas. Cuenta con un grupo de técnicos y tiene una estrategia de trabajo algo particular. Cuando alguien solicita un técnico, si alguno de sus técnicos se encuentra en la empresa, se lo envía inmediatamente al domicilio de la persona. En caso de no haber técnicos disponibles, el pedido queda *pendiente* a la espera de que algún técnico se desocupe. Cuando un técnico termina de resolver un problema, si hay pedidos *pendientes*, la central le asigna al técnico el pedido pendiente más cercano al domicilio en el que éste se encuentra y el técnico se dirige automáticamente hacia allí. Por el contrario, de no haber trabajos pendientes, el técnico regresa a la central y queda disponible para futuros trabajos. Dada una dirección, el TAD especificado permite saber quiénes fueron los técnicos que la visitaron la mayor cantidad de veces (aun si todavía se encuentran en la misma). El siguiente TAD modela la empresa *TaD* (aunque se omiten las axiomatizaciones).

TAD técnico es nat

TAD TAD

géneros tad

observadores básicos

libres : tad \rightarrow conj(técnico)

ocupados : tad \rightarrow conj(técnico)

ubicación : tad $s \times$ técnico $t \rightarrow$ dirección

pendientes : tad \rightarrow secu(dirección)

visitas : tad $s \times$ técnico $t \times$ dirección $d \rightarrow$ nat

$\{ t \in \text{ocupados}(s) \}$

$\{ t \in \text{ocupados}(s) \cup \text{libres}(s) \}$

generadores

iniciar : conj(técnico) \rightarrow tad

solicitar : tad \times dirección \rightarrow tad

finalizar : tad $s \times$ técnico $t \rightarrow$ tad

$\{ t \in \text{ocupados}(s) \}$

otras operaciones

másLaVisitaron : tad \times dirección \rightarrow conj(técnico)

...

Fin TAD

Se decidió utilizar la siguiente estructura de representación:

TAD se representa con estr

donde estr es tupla \langle técnicos: conj(técnico),
 clientes: conj(dirección),
 ubicación: dicc(técnico, dirección),
 quienesEstánEn: dicc(dirección, conj(técnico)),
 pendientes: secu(dirección),
 visitas: dicc(técnico, multiconj(dirección)) \rangle

En esta estructura, *técnicos* son los técnicos de la empresa y *clientes* guarda todas las direcciones alguna vez visitadas. Por otro lado, *ubicación* indica en dónde se encuentran los técnicos que están actualmente ocupados y *quienesEstánEn* registra los técnicos que están trabajando en una dirección dada. A su vez, *pendientes* guarda la secuencia de pedidos pendientes. Finalmente, para cada técnico t , *visitas* guarda un multiconjunto con todas las direcciones que visitó t (aun si está actualmente en la misma).

Teniendo en cuenta el TAD presentado arriba y la estructura elegida para su representación se pide:

- Escribir en castellano el invariante de representación.
- Escribir formalmente el invariante de representación.
- Escribir formalmente la función de abstracción.

1	2	3
P	P	P

(P)

1

① TAD CCC

Observadores básicos

Fig. Obs (última hoja)

- ✓ Maestros: $CCC \subset \rightarrow \text{conf}(\text{persona})$
- ✓ directores: $CCC \subset \rightarrow \text{conf}(\text{persona})$
- ✓ estudiantes: $CCC \subset \rightarrow \text{conf}(\text{persona})$
- ✓ aprendices de ~~maestro~~: $CCC \subset \times \text{persona } p \rightarrow \text{conf}(\text{persona})$
- ✓ Nivel De: $CCC \subset \times \text{persona } p \rightarrow \text{Nat}$

$\{p \in \text{estudiantes}(c) \cup$
 $\text{Maestros}(c)\}$
 $\{p \in \text{estudiantes}(c)\}$

Generadores

- ✓ Iniciar: $\text{conf}(\text{persona}) \subset \rightarrow CCC$ $\{ \emptyset?(c) \}$
- ✓ Ingresar: $CCC \subset \times \text{persona } p \rightarrow CCC$ $\{ p \notin (\text{maestros}(c) \cup \text{estudiantes}(c) \cup \text{directores}(c)) \}$
- ✓ Desetar: $CCC \subset \times \text{persona } p \rightarrow CCC$ $\{ ((p \in \text{maestros}(c) \wedge \# \text{maestros}(c) > 1) \vee (p \in \text{maestros}(c) \cup \text{estudiantes}(c))) \}$
- ✓ sig Nivel: $CCC \subset \times \text{persona } p \rightarrow CCC$ $\{ p \in \text{estudiantes}(c) \wedge p \notin \text{maestros}(c) \}$

Otras Operaciones

descendientesDe : $ccc \subset \times persona\ p \rightarrow conf(persona) \setminus \{p \in esclavos(c) \cup maestros(c)\}$

tutorDe : $ccc \subset \times persona\ p \rightarrow persona \setminus \{p \in esclavos(c)\}$

reconstruyeDescendientesDe : $ccc \subset \times conf(persona)\ cp \rightarrow conf(persona)$

apoDeapo : $ccc \subset \times conf(persona)\ cp \rightarrow conf(persona)$

tutorDeAux : $ccc \subset \times persona\ p \times conf(persona)\ cp \rightarrow persona$
 $\{ \text{~~persona~~ } \}$
 $\neg \exists (q) \wedge p \in esclavos(c)$

Axiomas

$$\bullet \text{maestros}(\text{Iniciar}(c)) \equiv c$$

$$\text{maestros}(\text{Ingresar}(c, p)) \equiv \text{maestros}(c)$$

$$\text{maestros}(\text{Desertar}(c, p)) \equiv \text{maestros}(c) \setminus (\text{if } p \in \text{maestros}(c) \text{ then } \{p\} \text{ else } \emptyset)$$

$$\text{maestros}(\text{sigNivel}(c, p)) \equiv \text{maestros}(c) \cup (\text{if } \text{nivelDe}(c, p) = 10 \text{ then } \{p\} \text{ else } \emptyset)$$

$$\bullet \text{desertores}(\text{Iniciar}(c)) \equiv \emptyset$$

$$\text{desertores}(\text{Ingresar}(c, p)) \equiv \text{desertores}(c)$$

$$\text{desertores}(\text{Desertar}(c, p)) \equiv \text{desertores}(c) \cup \{p\}$$

$$\text{desertores}(\text{sigNivel}(c, p)) \equiv \text{desertores}(c)$$

$$\bullet \text{esclavos}(\text{Iniciar}(c)) \equiv \emptyset$$

$$\text{esclavos}(\text{Ingresar}(c, p)) \equiv \text{esclavos}(c) \cup \{p\}$$

$$\text{esclavos}(\text{Desertar}(c, p)) \equiv \text{esclavos}(c) \setminus (\text{if } p \in \text{esclavos}(c) \text{ then } \{p\} \text{ else } \emptyset)$$

$estudantes(sigNivel(c, p)) \equiv \text{estudantes}(c) \setminus (\text{if } nivelDe(c, p) = 10$
 $\text{then } \{p\} \text{ else } \emptyset)$

• $aprendicesDe(Trinar(c), p) \equiv \emptyset$

$aprendicesDe(Ingresa(c, p'), p) \equiv \text{if } p = p' \text{ then } \{p\} \text{ else } \emptyset$

$\text{if } p = p' \text{ then}$

\emptyset

else

$\text{if } (nivelDe(c, p) > 1 \wedge \text{dondeVivo(estudiantes}(c) \cup \text{maestros}(c)) = p) \text{ then}$

$aprendicesDe(c, p) \cup \{p\}$

else

$aprendicesDe(c, p)$

fi

fi

$aprendicesDe(Desertar(c, p'), p) \equiv \text{if } p' \text{ es hoja 3}$
 ~~$\text{if } p' \text{ es hoja 3 then } \{p\} \text{ else } \emptyset$~~

$aprendicesDe(sigNivel(c, p'), p) \equiv \text{if } p' \in \text{aprendicesDe}(c, p) \text{ then } \{p\} \text{ else } \emptyset$

$\text{if } (p' \in \text{aprendicesDe}(c, p) \wedge nivelDe(c, p') =$
 $nivelDe(c, p) - 1) \text{ then}$

$\{p\} \cup \text{aprendicesDe}(c, p) \setminus \{p'\}$

else

$\text{if } (nivelDe(c, p') = nivelDe(c, \text{tutorDe}(c, p')) - 1$

$\wedge \text{dondeVivo(estudiantes}(c) \cup \text{maestros}(c)) = p \wedge$

$nivelDe(sigNivel(c, p'), p') < nivelDe(c, p) \text{ then}$


```

*  $\text{aprendicesDe}(\text{Detector}(c, p'), p) \equiv$  if  $p' \in \text{aprendicesDe}(c, p)$  then
     $\text{aprendicesDe}(c, p) \setminus \{p'\}$ 
    else
    if  $p = p' + tw$ 
    then
    else
    if  $(\text{doublet}(\text{estudiantes}(\text{Detector}(c, p')) \cup$ 
     $\text{nostras}(\text{Detector}(c, p')))) = p$ 
    and  $p' \in \text{nostras}(c)$ 
    then
     $\text{aprendicesDe}(c, p) \cup$ 
     $\text{aprendicesDe}(c, p')$ 
    else
    return
    if error  $(p' \notin \text{nostras}(c))$ 
    then  $\text{tutorDe}(c, p') = p$ 
    then
     $\text{aprendicesDe}(c, p) \cup$ 
     $\text{aprendicesDe}(c, p')$ 
    else
     $\text{aprendicesDe}(c, p)$ 
    fi
    fi
    fi

```

Igualdad Observacional

$$\begin{aligned} (\forall c_1, c_2: \text{ccc}) \quad c_1 \approx_{\text{obs}} c_2 & \Leftrightarrow \text{maestros}(c_1) = \text{maestros}(c_2) \wedge \\ & \text{directores}(c_1) = \text{directores}(c_2) \wedge \\ & \text{estudiantes}(c_1) = \text{estudiantes}(c_2) \wedge \\ & (\forall p: \text{persona}) \quad (p \in \text{estudiantes}(c_1) \vee \\ & \text{maestros}(c_1) \Rightarrow \neg \\ & \text{aprendicesDe}(c_1, p) = \text{aprendicesDe}(c_2, p)) \\ & (\forall p: \text{persona}) \quad (p \in \text{estudiantes}(c_1) \Rightarrow \neg \\ & \text{nivelDe}(c_1, p) = \text{nivelDe}(c_2, p)) \end{aligned}$$

persona es NOT

$$*2 \text{ nivelDe}(\text{Ingresor}(c, p'), p) \equiv \text{if } p = p' \text{ then } 1 \text{ else } \text{nivelDe}(c, p)$$

$$\text{nivelDe}(\text{SigNivel}(c, p'), p) \equiv \text{nivelDe}(c, p) + (\text{if } p = p' \text{ then } 1 \text{ else } 0)$$

✓
impeccable!

② 1) si $f(n) \in O(n) \Rightarrow 2^{f(n)} \in O(2^n)$

$f(n) \in O(n) \Leftrightarrow \exists c, n_0 / \forall n \geq n_0 \quad f(n) \leq c \cdot n$

$2^{f(n)} \in O(2^n) \Leftrightarrow \exists c', n'_0 / \forall n \geq n'_0 \quad 2^{f(n)} \leq c' \cdot 2^n$

Tomando $f(n) = 2n$ y $2n \in O(n)$ tomando $c=2$ y $n_0=1$
ya que $\forall n \geq n_0 \quad 2n \leq 2n$

Veremos que $\nexists c', n'_0 / 2^{2n} \in O(2^n)$

si: $2^{f(n)} \in O(2^n) \quad 2^{2n} \leq c' \cdot 2^n$

$2^n \leq c'$ pero $2^n \xrightarrow{n \rightarrow \infty} \infty$ y c' es una constante, luego absurdo.

FALSO.

2) $f \cdot g / (f \cdot g)(x) = f(x)g(x) \Rightarrow f \cdot g \in O(\max\{f, g\}^2)$

$f \cdot g \in O(\max\{f, g\}^2) \Leftrightarrow \exists c', n'_0 / \forall n \geq n'_0$

$c' \max\{f, g\}^2 \leq f \cdot g \leq c \cdot \max\{f, g\}^2$

Tomando $f(n) = n$ y $g(n) = n^2$ $\max\{f, g\}^2 = (n^2)^2 = n^4$

$f \cdot g = n^3$

$c' n^4 \leq n^3 \leq c n^4$

$\nexists c' n^4 \leq n^3$

$c' n \leq 0$ pero $c' n \rightarrow \infty$ $n \rightarrow \infty$

luego absurdo.

FALSO

~~El algoritmo no depende de la propiedad de la entrada (si es par/impar por ejemplo), depende solo de la magnitud de n . La complejidad de pases del algoritmo es: $T(n) = n + 100n + O(1) = 101n + O(1) = \Theta(n)$ ya que el primer ciclo hace n iteraciones y se entra en el segundo ciclo de n iteraciones 100 veces.~~

3/4) Este algoritmo no depende de la propiedad de la entrada (si n es par/impar por ejemplo), depende solo de la magnitud ~~de~~ n . La complejidad de pases del algoritmo es: $T(n) = n + 100n + O(1) = 101n + O(1) = \Theta(n)$ ya que el primer ciclo hace n iteraciones y se entra en el segundo ciclo de n iteraciones 100 veces.

Por lo tanto (3) es FALSO ya que $\nexists c, m_0 / cm^2 \leq n \forall n \geq m_0$ y (4) es VERDADERO ya que $\Theta(n) \subseteq O(n)$.

- técnicos es igual a claves de visitas
- ③ a) (1) ~~técnicos es igual a los datos de quienes están~~
(2) las claves de ubicación están contenidas en técnicos
(3) toda dirección en clientes fue visitada por un técnico y toda dirección visitada por un técnico está en clientes
(4) clientes y pendientes son disjuntos (no heredidad)
(5) La unión de clientes y pendientes es igual a ~~los~~ claves de quienes están en
(6) ~~toda~~ Para todo técnico definido en ubicación su dirección está definido en quienes están en y este técnico pertenece al significado de la dirección. Lo mismo para toda dirección definida en quienes están en y los técnicos en su significado.
- ~~Nada~~ ~~toda dirección pertenece~~

y visitas? ah perdón! es ③ ✓

b) * Rep: $estr \ e \rightarrow \text{bool}$

$(\forall e. estr) \text{ Rep}(e) = \text{true} \Leftarrow$

(1) $e.\text{técnicos} = \text{claves}(e.\text{visitas}) \wedge$

(2) $\text{claves}(e.\text{ubicación}) \in e.\text{técnicos} \wedge$

(3) ~~$(\forall d: \text{dirección}) (\exists t: \text{técnico}) (d \in \text{claves}(t.\text{visitas}) \Rightarrow$~~
 ~~$(\exists t: \text{técnico}) (\text{def?}(t, e.\text{ubicación}) \Rightarrow$~~
 ~~$\text{Obtener}(t, e.\text{ubicación}) = d \wedge d \in \text{Obtener}(t, e.\text{visitas})) \wedge$~~

$(\forall t: \text{técnico}) (\text{def?}(t, e.\text{ubicación}) \Rightarrow \text{Obtener}(t, e.\text{ubicación}) \in$
 $e.\text{clientes} \wedge (\forall d: \text{dirección}) (d \in \text{Obtener}(t, e.\text{visitas}) \Rightarrow$
 $d \in e.\text{clientes})) \wedge$

(4) $(\forall d: \text{dirección}) (d \in e.\text{clientes} \Rightarrow \text{está?}(d, e.\text{puddles})) \wedge$
 $(\forall d: \text{dirección}) (\text{está?}(d, e.\text{puddles}) \Rightarrow d \in e.\text{clientes})$

(5) $(\forall d: \text{dirección}) (\text{está?}(d, e.\text{puddles}) \vee d \in e.\text{clientes} \Rightarrow$
 $d \in \text{claves}(e.\text{quienesEstánEn})) \wedge$

$(\forall d: \text{dirección}) (d \in \text{claves}(e.\text{quienesEstánEn}) \Rightarrow$
 $\text{está?}(d, e.\text{puddles}) \vee d \in e.\text{clientes}) \wedge$

(6) $(\forall t: \text{técnico}) (\text{def?}(t, e.\text{ubicación}) \Rightarrow$
 ~~$t \in \text{Obtener}(\text{Obtener}(t, e.\text{ubicación}), e.\text{quienesEstánEn})$~~ $) \wedge$

$(\forall d: \text{dirección}) (\text{def?}(d, e.\text{quienesEstánEn}) \Rightarrow$

$(\forall t: \text{técnico}) (t \in \text{Obtener}(d, e.\text{quienesEstánEn}) \Rightarrow$
 $\text{def?}(t, e.\text{ubicación}) \wedge \text{Obtener}(t, e.\text{ubicación}) = d)$

g) Abs: $\text{estr } e \rightarrow \text{ToD}$

$\{ \text{Rep}(e) \}$

$\text{Abs}(e) \equiv \text{is} / e.\text{tecnicos} \setminus \text{claves}(e.\text{ubicacion}) = \text{libres}(\text{is}) \wedge$

~~is~~ $\text{claves}(e.\text{ubicacion}) = \text{ocupados}(\text{is}) \wedge$

$(\forall t: \text{tecnico}) (t \in \text{ocupados}(\text{is}) \Rightarrow$

$\text{obtener}(t, e.\text{ubicacion}) = \text{ubicacion}(s, t) \wedge$

$e.\text{pendientes} = \text{pendientes}(s) \wedge$

$(\forall t: \text{tecnico}) (T \in e.\text{tecnicos} \Rightarrow$

$(\forall d: \text{direccion}) \text{contAplicaciones}(d, \text{obtener}(t, e.\text{visitas})) =$
 $\text{visitas}(s, t, d))$

contAplicaciones: $\text{direccion } d \times \text{multiconj}(\text{direccion}) n \rightarrow \text{nat}$

$\text{contAplicaciones}(d, n) \equiv \text{if } \emptyset?(n) \text{ then}$

0

else

if $\text{dom}(\text{hd}(n)) = d$ then

$1 + \text{contAplicaciones}(d, \text{simhd}(n))$

else

$\text{contAplicaciones}(d, \text{simhd}(n))$

fi fi

↓
 no hace
 falta
 en multiconj
 pues #